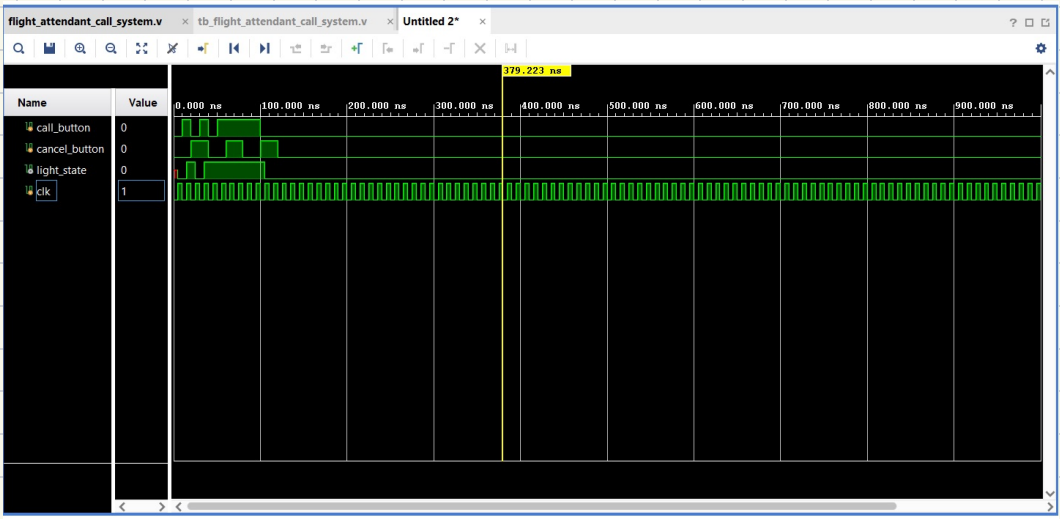
Lab 4 Report

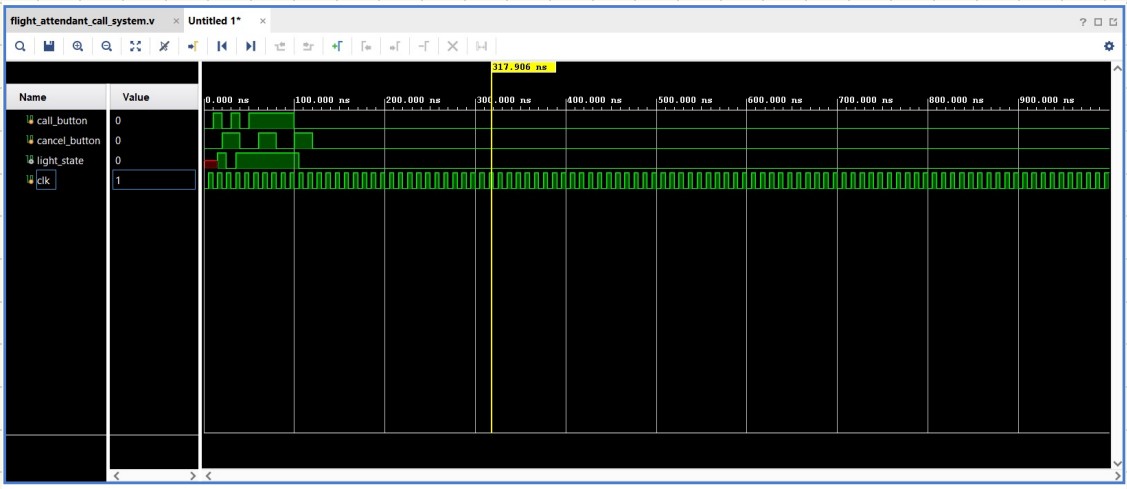
By Niranjan Telkikar and Ryan Mayeux
EID: rm63872 & nnt479

Part 1

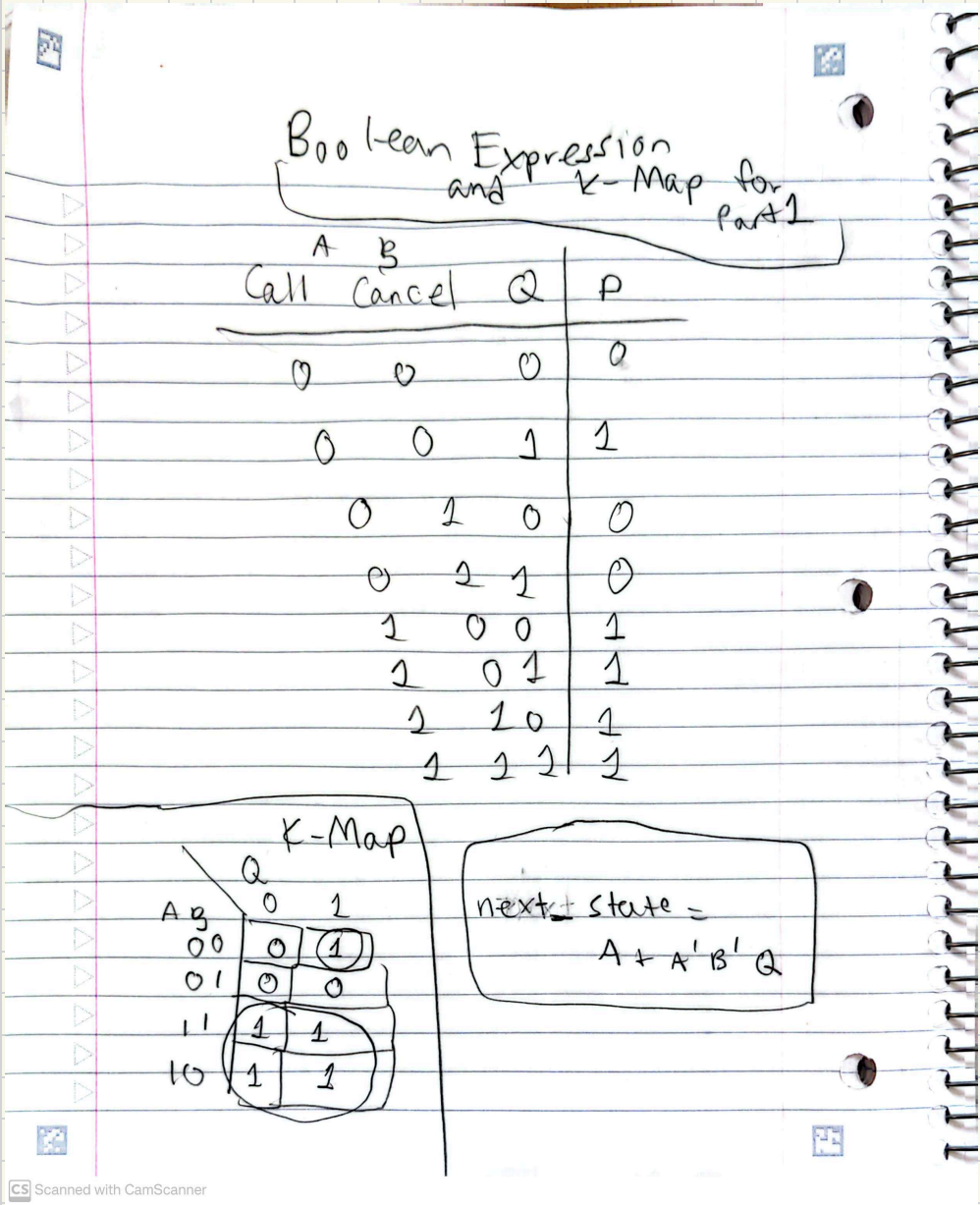
i. Simulation waveform of Flight Attendant Call System (Behavioral)



i (b). Simulation Waveform (Dataflow Modelling)



ii. KMap for Minimizing Expression for next_state
iii. Boolean expression for next_state for dataflow modelling

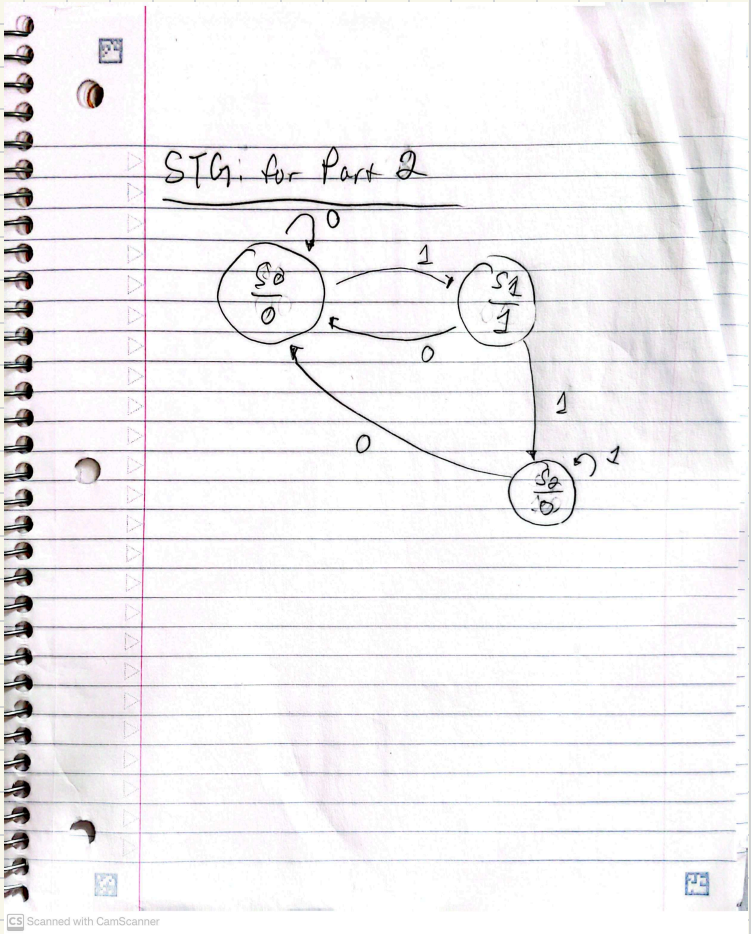


iv. Design file for dataflow modelling

```
`timescale 1ns / 1ps

module flight_attendant_call_system(
    input wire clk,
    input wire call_button,
    input wire cancel_button,
    output reg light_state
);

    wire next_state;

    // Combinational block
//    always @(*) begin
//        case ({call_button, cancel_button, light_state})
//            3'b000: next_state = 1'b0;
//            3'b001: next_state = 1'b1;
//            3'b010: next_state = 1'b0;
//            3'b011: next_state = 1'b0;
//            3'b100: next_state = 1'b1;
//            3'b101: next_state = 1'b1;
//            3'b110: next_state = 1'b1;
//            3'b111: next_state = 1'b1;
//            default: next_state = 1'b0;
//        endcase
//    end
    assign next_state = call_button | (~call_button & ~cancel_button & light_state);

    // Sequential block
    always @(posedge clk) begin
        light_state <= next_state;
    end

endmodule
```

Part 2

v. State Diagram of Rising Edge Detector



vi. Completed design files including top module and clock divider

```
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module rising_edge_detector(
input clk,
input signal,
input reset,
output reg outedge
    );

wire slow_clk;

reg [1:0] state;
reg [1:0] next_state;

clkdiv c1(clk, reset, slow_clk);
//assign slow_clk = clk;  // use fast clock for simulation


always @(*) begin
```

```
always @(*) begin
    case (state)
    2'b00 : begin
         outedge = 1'b0;
         if(~signal)
           next_state=2'b00;
         else
           next_state=2'b01;
        end

        2'b01: begin outedge = 1'b1;
         if(~signal)
           next_state=2'b00;
         else
           next_state=2'b10;
        end


        2'b10: begin outedge = 1'b0;
        if(~signal)
           next_state=2'b00;
         else
```

```
          next_state=2'b10;
       end


       default: begin
            next_state = 2'b00;
            outedge= 1'b0;
            end
        endcase
     end


always@(posedge slow_clk or posedge reset) begin
if(reset)
    state <= 2'b00;
else
    state <= next_state;
end
endmodule
```

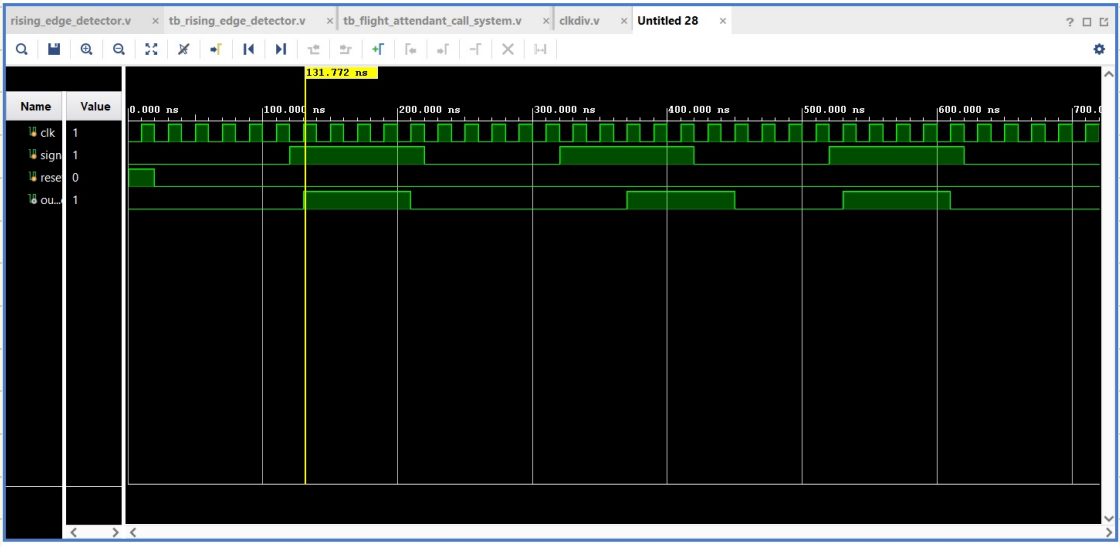
```
13      //
14      // Dependencies:
15      //
16      // Revision:
17      // Revision 0.01 - File Created
18      // Additional Comments:
19      //
20      //////////////////////////////////////////////////////////////////////////////////


23      module clkdiv(
24      input clk,
25      input reset,
26      output clk_out
27          );
28
29      reg [25:0] COUNT;
30
31      assign clk_out = COUNT[25];
32
33      always @(posedge clk)
34      begin
35      if(reset)
36          COUNT=0;
37      else
38          COUNT=COUNT+1;
39      end
40
41      endmodule
```

vii. Test Bench of System

```
module tb_rising_edge_detector;

reg clk;
reg signal;
reg reset;
wire outedge;


    rising_edge_detector DUT (
        .clk(clk),
        .signal(signal),
        .reset(reset),
        .outedge(outedge)
    );

// Instantiate clkdiv

    always #10 clk = ~clk;  // 10 ns period = 100 MHz

 initial begin
 clk = 0;
 signal = 0;
```

```
43    clk = 0;
44    signal = 0;
45    reset=1;
46    #20
47    reset=0;
48    #100
49    signal=1;
50    #100
51    signal=0;
52    #100
53    signal=1;
54    #100
55    signal=0;
56    #100
57    signal=1;
58    #100
59    signal=0;
60    #100 $finish;
61    end
62
63  endmodule
64
```

viii. Simulation Waveform



ix. Constraints File

```
## Clock signal - Uncomment if needed (will be used in future labs)
set_property PACKAGE_PIN W5 [get_ports {clk}]
    set_property IOSTANDARD LVCMOS33 [get_ports {clk}]
    create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {clk}]

# Switches
## Connects pin V17 (SW0 on the board) to input a in our gate module
set_property PACKAGE_PIN V17 [get_ports {signal}]
## Sets the switch to use 3.3V logic
    set_property IOSTANDARD LVCMOS33 [get_ports {signal}]

### Connects pin V16 (SW1 on the board) to input b in our gate module
#set_property PACKAGE_PIN V16 [get_ports {b}]
```

```
## LEDs
## Connects the output c in our gate module to pin U16 (LED0 on-board)
set_property PACKAGE_PIN U16 [get_ports {outedge}]
## Sets the LED to use 3.3V logic
    set_property IOSTANDARD LVCMOS33 [get_ports {outedge}]
```

```
##Buttons
set_property PACKAGE_PIN U18 [get_ports {reset}]
    set_property IOSTANDARD LVCMOS33 [get_ports {reset}]
#set_property PACKAGE_PIN T18 [get_ports btnU]
    #set_property IOSTANDARD LVCMOS33 [get_ports btnU]
#set_property PACKAGE_PIN W19 [get_ports {call_button}]
```

Part 3

x. Completed Design Files of all Modules in the System

```
`timescale 1ns / 1ps

module time_multiplexing_main(
    input clk,
    input reset,
    input [15:0] sw,
    output [3:0] an,
    output [6:0] sseg,
    output slow_clk,
    output [7:0] binary_out
);

wire [6:0] in0, in1, in2, in3;
assign binary_out = sw[7:0];

// Module instantiation of hexto7segment decoder
hexto7segment h0 (.x(sw[3:0]),   .r(in0));
hexto7segment h1 (.x(sw[7:4]),   .r(in1));
hexto7segment h2 (.x(sw[11:8]),  .r(in2));
hexto7segment h3 (.x(sw[15:12]), .r(in3));

// Module instantiation of clock divider
clkdiv c0(.clk(clk), .reset(reset), .clk_out(slow_clk));

// Module instantiation of the multiplexer
time_mux_state_machine f1_inst(
    .clk (slow_clk),
    .reset (reset),
    .in0 (in0),
    .in1 (in1),
    .in2 (in2),
    .in3 (in3),
    .an (an),
    .sseg (sseg)
);

endmodule
```

```
`timescale 1ns / 1ps

module time_mux_state_machine(
    input clk,
    input reset,
    input [6:0] in0,
    input [6:0] in1,
    input [6:0] in2,
    input [6:0] in3,
    output reg [3:0] an,
    output reg [6:0] sseg
);

reg [1:0] state;
reg [1:0] next_state;

//----------------------------------
// State transition logic
//----------------------------------
always @(*) begin
    case (state)
        2'b00: next_state = 2'b01; // Example
        2'b01: next_state = 2'b10;
        2'b10: next_state = 2'b11;
        2'b11: next_state = 2'b00;
    endcase
end

//----------------------------------
// Multiplexer logic
//----------------------------------
always @(*) begin
    case (state)
        2'b00: sseg = in0;
        2'b01: sseg = in1;
        2'b10: sseg = in2;
        2'b11: sseg = in3;
```

```
    //----------------------------------
    // Multiplexer logic
    //----------------------------------
    always @(*) begin
        case (state)
            2'b00: sseg = in0;
            2'b01: sseg = in1;
            2'b10: sseg = in2;
            2'b11: sseg = in3;
        endcase
    end

    //----------------------------------
    // Decoder (anode enable) logic
    //----------------------------------
    always @(*) begin
        case (state)
            2'b00: an = 4'b1110; // Example
            2'b01: an= 4'b1101;
            2'b10: an= 4'b1011;
            2'b11: an= 4'b0111;
        endcase
    end

    //----------------------------------
    // Sequential state register
    //----------------------------------
    always @(posedge clk or posedge reset) begin
        if (reset)
            state <= 2'b00;
        else
            state <= next_state;
    end

    endmodule
```

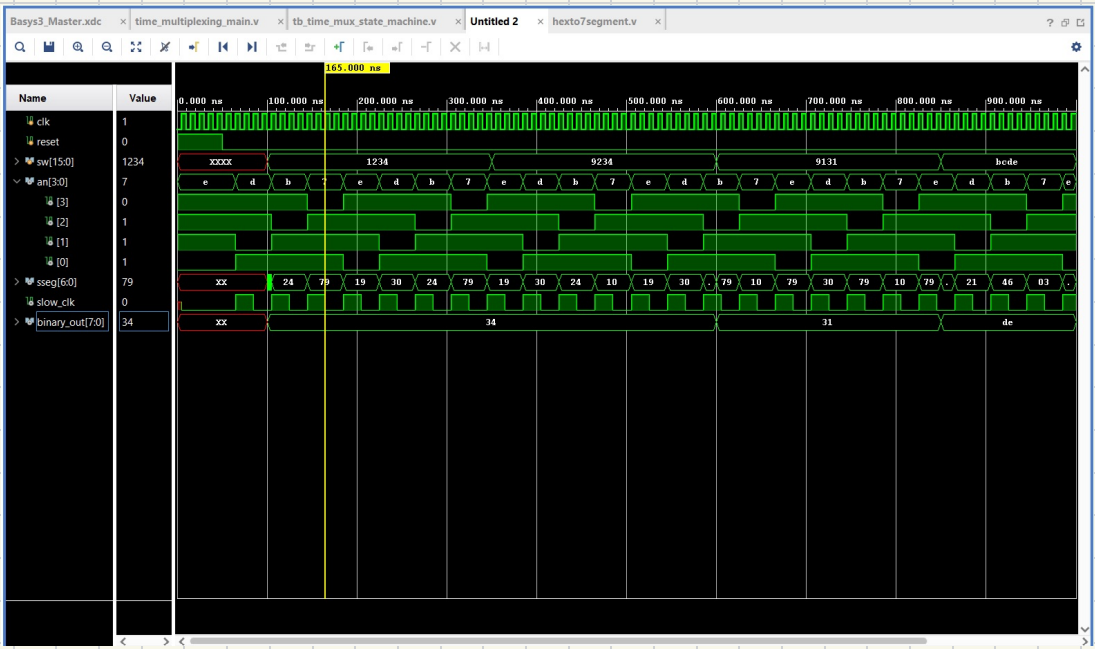
```
module hexto7segment(
input [3:0] x,
output reg [6:0] r
    );

always @(*)
    case(x)
        4'b0000: r = 7'b1000000; // 0
        4'b0001: r = 7'b1111001; // 1
        4'b0010: r = 7'b0100100; // 2
        4'b0011: r = 7'b0110000; // 3
        4'b0100: r = 7'b0011001; // 4
        4'b0101: r = 7'b0010010; // 5
        4'b0110: r = 7'b0000010; // 6
        4'b0111: r = 7'b1111000; // 7
        4'b1000: r = 7'b0000000; // 8
        4'b1001: r = 7'b0010000; // 9
        4'b1010: r = 7'b0001000; // A
        4'b1011: r = 7'b0000011; // b
        4'b1100: r = 7'b1000110; // C
        4'b1101: r = 7'b0100001; // d
        4'b1110: r = 7'b0000110; // E
        4'b1111: r = 7'b0001110; // F
        endcase
endmodule
```

xi. Test–bench of System

```
module tb_time_mux_state_machine;

reg clk;
reg reset;
reg [15:0] sw;
wire [3:0] an;
wire [6:0] sseg;
wire slow_clk;
wire [7:0] binary_out;



    time_multiplexing_main DUT (
        .clk(clk),
        .reset(reset),
        .sw(sw),
        .an(an),
        .sseg(sseg),
        .slow_clk(slow_clk),
        .binary_out(binary_out)
    );

always #5 clk = ~clk;


initial begin
clk=0;
reset=1;
#50
reset=0;
```

```
reset=1;
#50
reset=0;
#50
sw=16'h1234;
#250
sw=16'h9234;
#250
sw=16'h9131;
#250
sw=16'hbCdE;
#550
$stop;

end
endmodule
```

xii. Simulation Waveform



xiii. Constraints file

```
## Clock signal - Uncomment if needed (will be used in future labs)
set_property PACKAGE_PIN W5 [get_ports {clk}]
    set_property IOSTANDARD LVCMOS33 [get_ports {clk}]
    create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {clk}]

# Switches
## Connects pin V17 (SW0 on the board) to input a in our gate module
set_property PACKAGE_PIN V17 [get_ports {sw[0]}]
## Sets the switch to use 3.3V logic
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]

### Connects pin V16 (SW1 on the board) to input b in our gate module
set_property PACKAGE_PIN V16 [get_ports {sw[1]}]
## Sets the switch to use 3.3V logic
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[1]}]

set_property PACKAGE_PIN W16 [get_ports {sw[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[2]}]
set_property PACKAGE_PIN W17 [get_ports {sw[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[3]}]
set_property PACKAGE_PIN W15 [get_ports {sw[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[4]}]
set_property PACKAGE_PIN V15 [get_ports {sw[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[5]}]
set_property PACKAGE_PIN W14 [get_ports {sw[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[6]}]
set_property PACKAGE_PIN W13 [get_ports {sw[7]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[6]}]
set_property PACKAGE_PIN W13 [get_ports {sw[7]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[7]}]
set_property PACKAGE_PIN V2 [get_ports {sw[8]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[8]}]
set_property PACKAGE_PIN T3 [get_ports {sw[9]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[9]}]
set_property PACKAGE_PIN T2 [get_ports {sw[10]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[10]}]
set_property PACKAGE_PIN R3 [get_ports {sw[11]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[11]}]
set_property PACKAGE_PIN W2 [get_ports {sw[12]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[12]}]
set_property PACKAGE_PIN U1 [get_ports {sw[13]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[13]}]
set_property PACKAGE_PIN T1 [get_ports {sw[14]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[14]}]
set_property PACKAGE_PIN R2 [get_ports {sw[15]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[15]}]


## LEDs
## Connects the output c in our gate module to pin U16 (LED0 on-board)
set_property PACKAGE_PIN U16 [get_ports {binary_out[0]}]
## Sets the LED to use 3.3V logic
    set_property IOSTANDARD LVCMOS33 [get_ports {binary_out[0]}]

set_property PACKAGE_PIN E19 [get_ports {binary_out[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {binary_out[1]}]
set_property PACKAGE_PIN U19 [get_ports {binary_out[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {binary_out[2]}]
set_property PACKAGE_PIN V19 [get_ports {binary_out[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {binary_out[3]}]
set_property PACKAGE_PIN W18 [get_ports {binary_out[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {binary_out[4]}]
set_property PACKAGE_PIN U15 [get_ports {binary_out[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {binary_out[5]}]
```

```
set_property PACKAGE_PIN U15 [get_ports {binary_out[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {binary_out[5]}]
set_property PACKAGE_PIN U14 [get_ports {binary_out[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {binary_out[6]}]
set_property PACKAGE_PIN V14 [get_ports {binary_out[7]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {binary_out[7]}]
#set_property PACKAGE_PIN V13 [get_ports {led[8]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {led[8]}]
#set_property PACKAGE_PIN V3 [get_ports {led[9]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {led[9]}]
#set_property PACKAGE_PIN W3 [get_ports {led[10]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {led[10]}]
#set_property PACKAGE_PIN U3 [get_ports {led[11]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {led[11]}]
#set_property PACKAGE_PIN P3 [get_ports {led[12]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {led[12]}]
#set_property PACKAGE_PIN N3 [get_ports {led[13]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {led[13]}]
#set_property PACKAGE_PIN P1 [get_ports {led[14]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {led[14]}]
set_property PACKAGE_PIN L1 [get_ports {slow_clk}]
    set_property IOSTANDARD LVCMOS33 [get_ports {slow_clk}]


#7 segment display
set_property PACKAGE_PIN W7 [get_ports {sseg[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[0]}]
set_property PACKAGE_PIN W6 [get_ports {sseg[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[1]}]
set_property PACKAGE_PIN U8 [get_ports {sseg[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[2]}]
set_property PACKAGE_PIN V8 [get_ports {sseg[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[3]}]
set_property PACKAGE_PIN U5 [get_ports {sseg[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[4]}]
set_property PACKAGE_PIN V5 [get_ports {sseg[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[5]}]
```

```
set_property PACKAGE_PIN V5 [get_ports {sseg[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[5]}]
set_property PACKAGE_PIN U7 [get_ports {sseg[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[6]}]

#set_property PACKAGE_PIN V7 [get_ports dp]
    #set_property IOSTANDARD LVCMOS33 [get_ports dp]

set_property PACKAGE_PIN U2 [get_ports {an[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {an[0]}]
set_property PACKAGE_PIN U4 [get_ports {an[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {an[1]}]
set_property PACKAGE_PIN V4 [get_ports {an[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {an[2]}]
set_property PACKAGE_PIN W4 [get_ports {an[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {an[3]}]


##Buttons
set_property PACKAGE_PIN U18 [get_ports {reset}]
    set_property IOSTANDARD LVCMOS33 [get_ports {reset}]
#set_property PACKAGE_PIN T18 [get_ports btnU]
    #set_property IOSTANDARD LVCMOS33 [get_ports btnU]
#set_property PACKAGE_PIN W19 [get_ports {call_button}]

#    set_property IOSTANDARD LVCMOS33 [get_ports {call_button}]

#set_property PACKAGE_PIN T17 [get_ports {cancel_button}]

#    set_property IOSTANDARD LVCMOS33 [get_ports {cancel_button}]

 #set_property PACKAGE_PIN U17 [get_ports btnD]
        #set_property IOSTANDARD LVCMOS33 [get_ports btnD]
```